

## Query 1 :

Create a stored procedure in the Northwind database that will calculate the average value of Freight for a specified customer. Then, a business rule will be added that will be triggered before every Update and Insert command in the Orders controller, and will use the stored procedure to verify that the Freight does not exceed the average freight. If it does, a message will be displayed and the command will be cancelled.

```
create procedure avgfreightbycustkvl(@CustId varchar(10))
```

```
as
```

```
begin
```

```
    select CustomerID, AVG(Freight) as [Average Freight]
```

```
    from Orders
```

```
    where CustomerID=@CustId
```

```
    group by CustomerID
```

```
end
```

```
exec avgfreightbycustkvl 'VINET'
```

```
create procedure insertintoorderskvl(@OrderID int, @CustomerID nchar(5),
```

```
    @EmployeeID int, @OrderDate datetime, @RequiredDate datetime,
```

```
    @ShippedDate datetime, @ShipVia int, @Freight money, @ShipName nvarchar(40),
```

```
    @ShipAddress nvarchar(40), @ShipCity nvarchar(40), @ShipRegion nvarchar(40),
```

```
    @ShipPostalCode nvarchar(40), @ShipCountry nvarchar(40))
```

```
as
```

```
begin
```

```
    declare @avgfreight money, @newCustomerID nchar(5)
```

```
    set @newCustomerID = @CustomerID
```

```
    set @avgfreight = (select AVG(Orders.Freight) from Orders where CustomerID =
```

```
    @CustomerID)
```

```
    if(@avgfreight>@Freight)
```

```
    begin
```

```
        INSERT INTO Orders(OrderID, CustomerID, EmployeeID, OrderDate,  
RequiredDate, ShippedDate, ShipVia, Freight, ShipName, ShipAddress, ShipCity,  
ShipRegion, ShipPostalCode, ShipCountry)
```

```
        VALUES (@OrderID, @CustomerID, @EmployeeID, @OrderDate,  
@RequiredDate, @ShippedDate, @ShipVia, @Freight, @ShipName, @ShipAddress,  
@ShipCity, @ShipRegion, @ShipPostalCode, @ShipCountry)
```

```
    end
```

```
    else
```

```
    begin
```

```

        RAISERROR ('Inserted Freight is more than the average freight of Customer'
, 10, 1)
    ROLLBACK TRANSACTION
end
END

```

```

SET IDENTITY_INSERT Orders ON

```

```

exec insertintoorderskvl '10246', 'VINET', '5', '', '', '1', '10', '', '',
'', ''

```

```

create procedure updateorderskvl(@OrderID int, @CustomerID varchar(10),
@EmployeeID int, @OrderDate datetime, @RequiredDate datetime, @ShippedDate
datetime, @ShipVia int, @Freight money, @ShipName nvarchar(40), @ShipAddress
nvarchar(40), @ShipCity nvarchar(40), @ShipRegion nvarchar(40), @ShipPostalCode
nvarchar(40), @ShipCountry nvarchar(40))
as
begin
    declare @avgfreight money, @newCustomerID nchar(5)
    set @newCustomerID = @CustomerID
    set @avgfreight = (select AVG(Orders.Freight) from Orders where CustomerID =
@CustomerID)

    if(@avgfreight>@Freight)
    begin
        update Orders
        set EmployeeID = @EmployeeID,
            OrderDate = @OrderDate,
            RequiredDate = @RequiredDate,
            ShippedDate = @ShippedDate,
            ShipVia = @ShipVia,
            Freight = @Freight,
            ShipName = @ShipName,
            ShipAddress = @ShipAddress,
            ShipCity = @ShipCity,
            ShipRegion = @ShipRegion,
            ShipPostalCode = @ShipPostalCode,
            ShipCountry = @ShipCountry
        where OrderID = @OrderID and CustomerID = @CustomerID
    end

    else
    begin
        RAISERROR ('updated Freight is more than the average freight of Customer'
, 10, 1)
        ROLLBACK TRANSACTION
    end
end

```

END                      end

```
exec updateorderskvl '10248', 'VINET', '5', ", ", ", ", '1', '12', ", ", ", ",
", "
', "
```

```
select * from Orders
where CustomerID = 'VINET'
```

```
exec insertintoorderskvl '10246', 'VINET', '5', '', '', '', '1', '10', '', '', ''
```

(1 row affected)

Completion time: 2023-02-08T14:40:32.0719719+05:30

```

begin
    RAISERROR ('updated Freight is more than the average freight of Customer'
    ROLLBACK TRANSACTION
end
END

exec updateorderskv1 '10248', 'VINET', '5', '', '', '', '1', '12', '', '', '', ''
', ''

select * from Orders
where CustomerID = 'VINET'

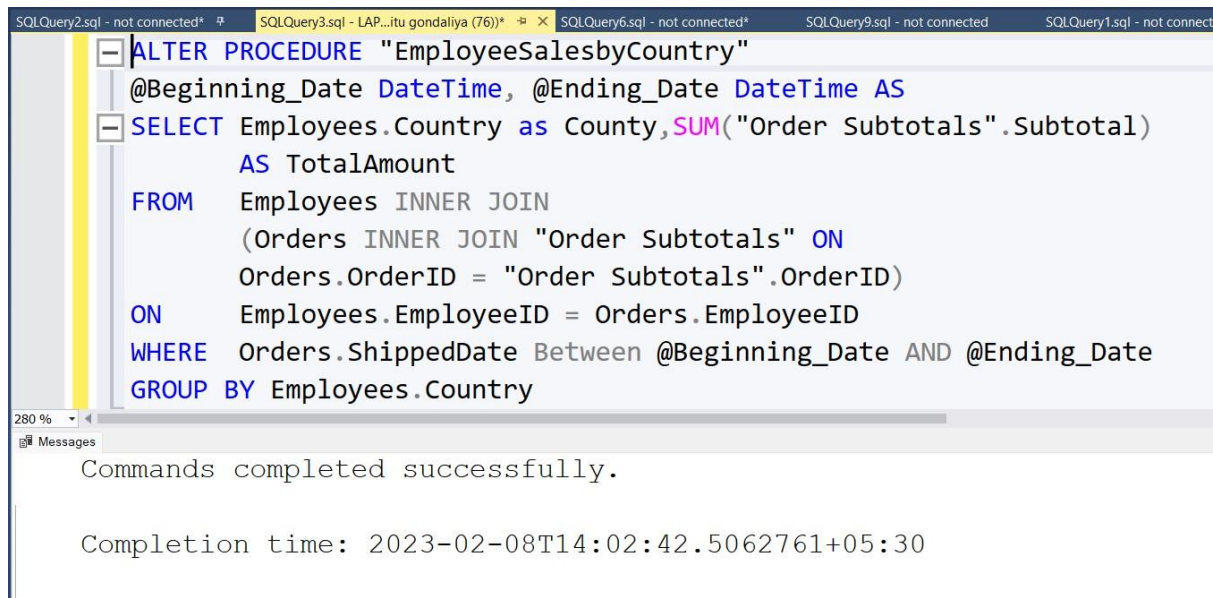
```

updated Freight is more than the average freight of Customer  
Msg 3903, Level 16, State 1, Procedure updateorderskv1, Line 30 [Batch Start L  
The ROLLBACK TRANSACTION request has no corresponding BEGIN TRANSACTION.

## Query 2 :

write a SQL query to Create Stored procedure in the Northwind database to retrieve Employee Sales by Country \*/

```
ALTER PROCEDURE "EmployeeSalesbyCountry"
@Beginning_Date DateTime, @Ending_Date DateTime AS
SELECT Employees.Country as County,SUM("Order Subtotals".Subtotal)
    AS TotalAmount
FROM  Employees INNER JOIN
    (Orders INNER JOIN "Order Subtotals" ON
        Orders.OrderID = "Order Subtotals".OrderID)
ON  Employees.EmployeeID = Orders.EmployeeID
WHERE Orders.ShippedDate Between @Beginning_Date AND @Ending_Date
GROUP BY Employees.Country
```



The screenshot displays the SQL Server Enterprise Manager interface. The top pane shows the SQL query for the stored procedure 'EmployeeSalesbyCountry'. The bottom pane shows the execution results, indicating that the commands were completed successfully and providing the completion time.

```
SQLQuery2.sql - not connected*  SQLQuery3.sql - LAP...itu gondaliya (76))  SQLQuery6.sql - not connected*  SQLQuery9.sql - not connected  SQLQuery1.sql - not connect
```

```
ALTER PROCEDURE "EmployeeSalesbyCountry"
@Beginning_Date DateTime, @Ending_Date DateTime AS
SELECT Employees.Country as County,SUM("Order Subtotals".Subtotal)
    AS TotalAmount
FROM  Employees INNER JOIN
    (Orders INNER JOIN "Order Subtotals" ON
        Orders.OrderID = "Order Subtotals".OrderID)
ON  Employees.EmployeeID = Orders.EmployeeID
WHERE Orders.ShippedDate Between @Beginning_Date AND @Ending_Date
GROUP BY Employees.Country
```

280 %

Messages

Commands completed successfully.

Completion time: 2023-02-08T14:02:42.5062761+05:30

EXECUTE EmployeeSalesbyCountry '1996-06-14', '1998-07-15'

SQLQuery2.sql - not connected\* SQLQuery3.sql - LAP...itu gondaliya (76))\* SQLQuery6.sql - not connected\* SQLQuery9.sql - not connected SQLQ...

EXECUTE EmployeeSalesbyCountry '1996-06-14', '1998-07-15'

280 %

Results Messages

	County	TotalAmount
1	USA	902466.39
2	UK	337244.44

### Query 3 :

write a SQL query to Create Stored procedure in the Northwind database to retrieve Sales by Year \*/

```
ALTER PROCEDURE "SalesbyYear"
```

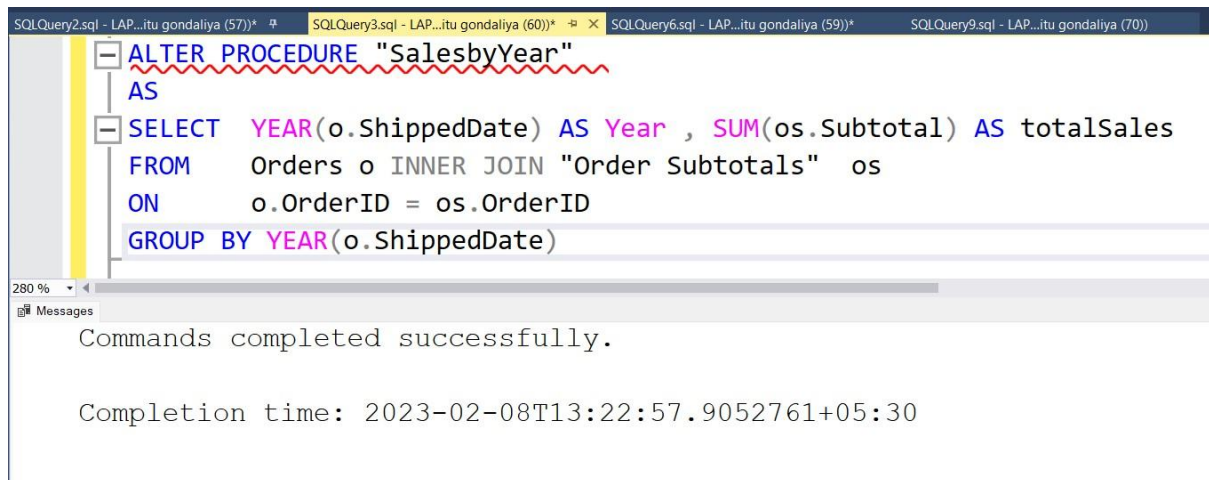
```
AS
```

```
SELECT YEAR(o.ShippedDate) AS Year , SUM(os.Subtotal) AS totalSales
```

```
FROM Orders o INNER JOIN "Order Subtotals" os
```

```
ON o.OrderID = os.OrderID
```

```
GROUP BY YEAR(o.ShippedDate)
```



```
SQLQuery2.sql - LAP...itu gondaliya (57))*  SQLQuery3.sql - LAP...itu gondaliya (60))*  SQLQuery6.sql - LAP...itu gondaliya (59))*  SQLQuery9.sql - LAP...itu gondaliya (70))
```

```
ALTER PROCEDURE "SalesbyYear"
AS
SELECT YEAR(o.ShippedDate) AS Year , SUM(os.Subtotal) AS totalSales
FROM Orders o INNER JOIN "Order Subtotals" os
ON o.OrderID = os.OrderID
GROUP BY YEAR(o.ShippedDate)
```

280 %

Messages

Commands completed successfully.

Completion time: 2023-02-08T13:22:57.9052761+05:30

EXECUTE SalesbyYear

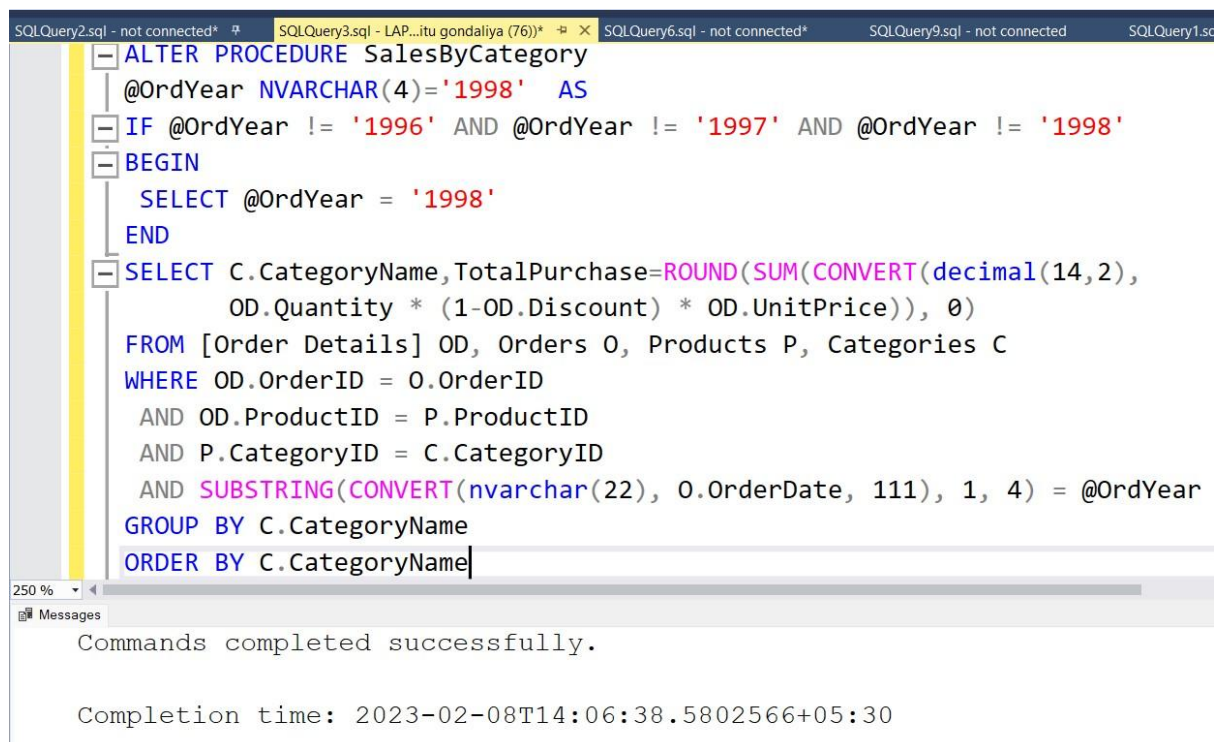
The screenshot shows a SQL query execution window with two tabs: 'SQLQuery2.sql - LAP...itu gondaliya (57))\*' and 'SQLQuery3.sql - LAP...itu gondaliya (57))\*'. The active tab displays the command 'EXECUTE SalesbyYear' in a large font. Below the command, there is a scroll bar and a zoom level of '280 %'. The 'Results' tab is selected, showing a table with four rows and two columns: 'Year' and 'totalSales'. The data is as follows:

	Year	totalSales
1	1998	437692.19
2	1996	193171.77
3	1997	608846.87
4	NULL	25937.43

#### Query 4 :

write a SQL query to Create Stored procedure in the Northwind database to retrieve Sales By Category \*/

```
ALTER PROCEDURE SalesByCategory
@OrdYear NVARCHAR(4)='1998' AS
IF @OrdYear != '1996' AND @OrdYear != '1997' AND @OrdYear != '1998'
BEGIN
    SELECT @OrdYear = '1998'
END
SELECT C.CategoryName, TotalPurchase=ROUND(SUM(CONVERT(decimal(14,2),
    OD.Quantity * (1-OD.Discount) * OD.UnitPrice)), 0)
FROM [Order Details] OD, Orders O, Products P, Categories C
WHERE OD.OrderID = O.OrderID
AND OD.ProductID = P.ProductID
AND P.CategoryID = C.CategoryID
AND SUBSTRING(CONVERT(nvarchar(22), O.OrderDate, 111), 1, 4) = @OrdYear
GROUP BY C.CategoryName
ORDER BY C.CategoryName
```



```
SQLQuery2.sql - not connected*  SQLQuery3.sql - LAP...itu gondaliya (76))*  SQLQuery6.sql - not connected*  SQLQuery9.sql - not connected  SQLQuery1.sql - not connected*

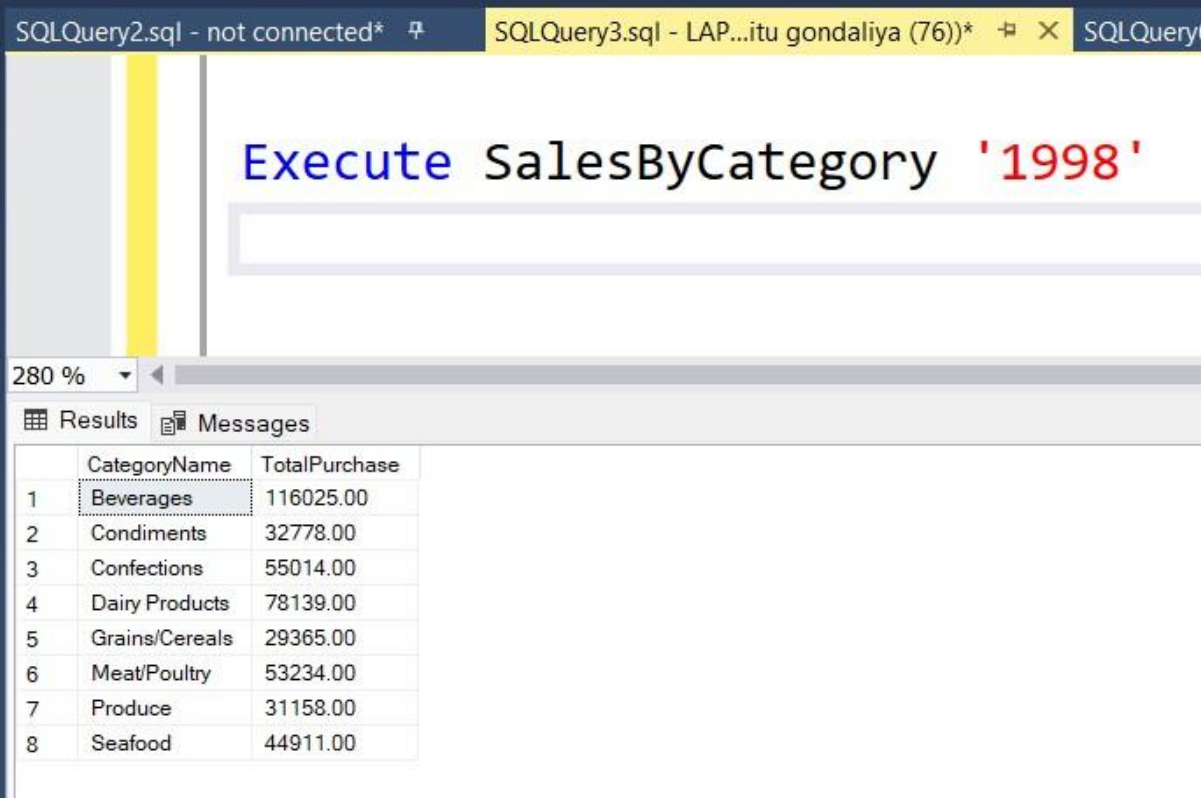
ALTER PROCEDURE SalesByCategory
@OrdYear NVARCHAR(4)='1998' AS
IF @OrdYear != '1996' AND @OrdYear != '1997' AND @OrdYear != '1998'
BEGIN
    SELECT @OrdYear = '1998'
END
SELECT C.CategoryName, TotalPurchase=ROUND(SUM(CONVERT(decimal(14,2),
    OD.Quantity * (1-OD.Discount) * OD.UnitPrice)), 0)
FROM [Order Details] OD, Orders O, Products P, Categories C
WHERE OD.OrderID = O.OrderID
AND OD.ProductID = P.ProductID
AND P.CategoryID = C.CategoryID
AND SUBSTRING(CONVERT(nvarchar(22), O.OrderDate, 111), 1, 4) = @OrdYear
GROUP BY C.CategoryName
ORDER BY C.CategoryName

250 %
Messages
Commands completed successfully.

Completion time: 2023-02-08T14:06:38.5802566+05:30
```



Execute SalesByCategory '1998'



SQLQuery2.sql - not connected\* SQLQuery3.sql - LAP...itu gondaliya (76))\* SQLQueryt

Execute SalesByCategory '1998'

280 %

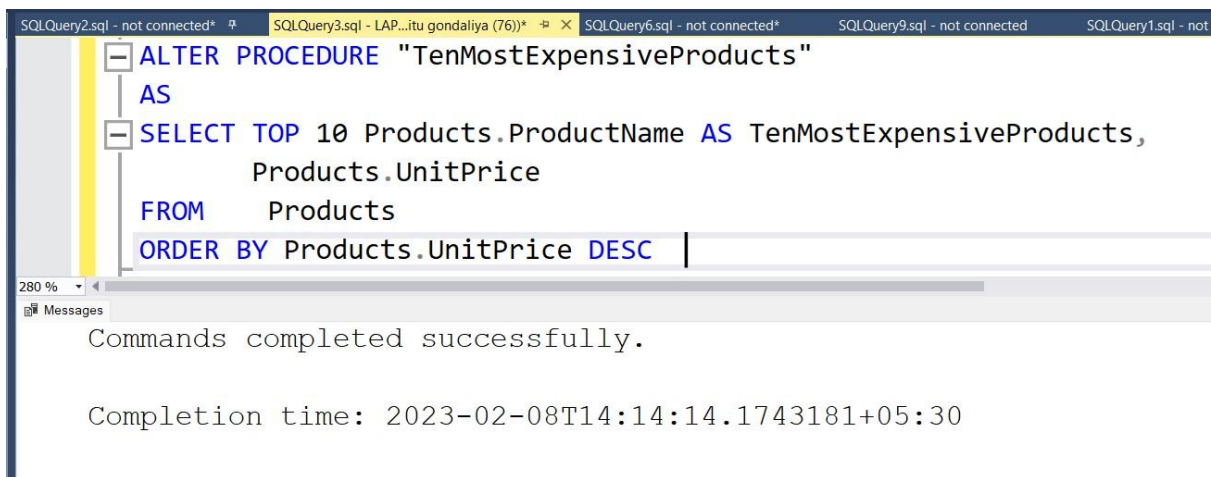
Results Messages

	CategoryName	TotalPurchase
1	Beverages	116025.00
2	Condiments	32778.00
3	Confections	55014.00
4	Dairy Products	78139.00
5	Grains/Cereals	29365.00
6	Meat/Poultry	53234.00
7	Produce	31158.00
8	Seafood	44911.00

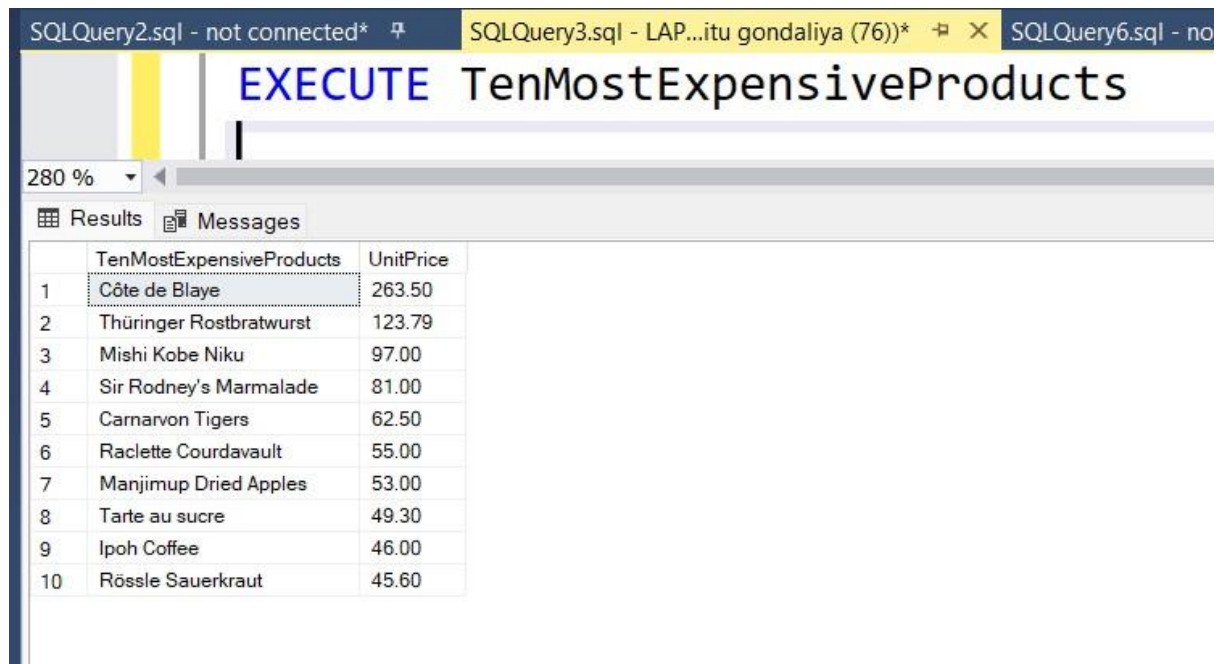
## Query 5 :

write a SQL query to Create Stored procedure in the Northwind database to retrieve Ten Most Expensive Products \*/

```
ALTER PROCEDURE "TenMostExpensiveProducts"  
AS  
SELECT TOP 10 Products.ProductName AS TenMostExpensiveProducts,  
       Products.UnitPrice  
FROM   Products  
ORDER BY Products.UnitPrice DESC
```



EXECUTE TenMostExpensiveProducts



SQLQuery2.sql - not connected\* SQLQuery3.sql - LAP...itu gondaliya (76))\* SQLQuery6.sql - no

**EXECUTE** TenMostExpensiveProducts

280 %

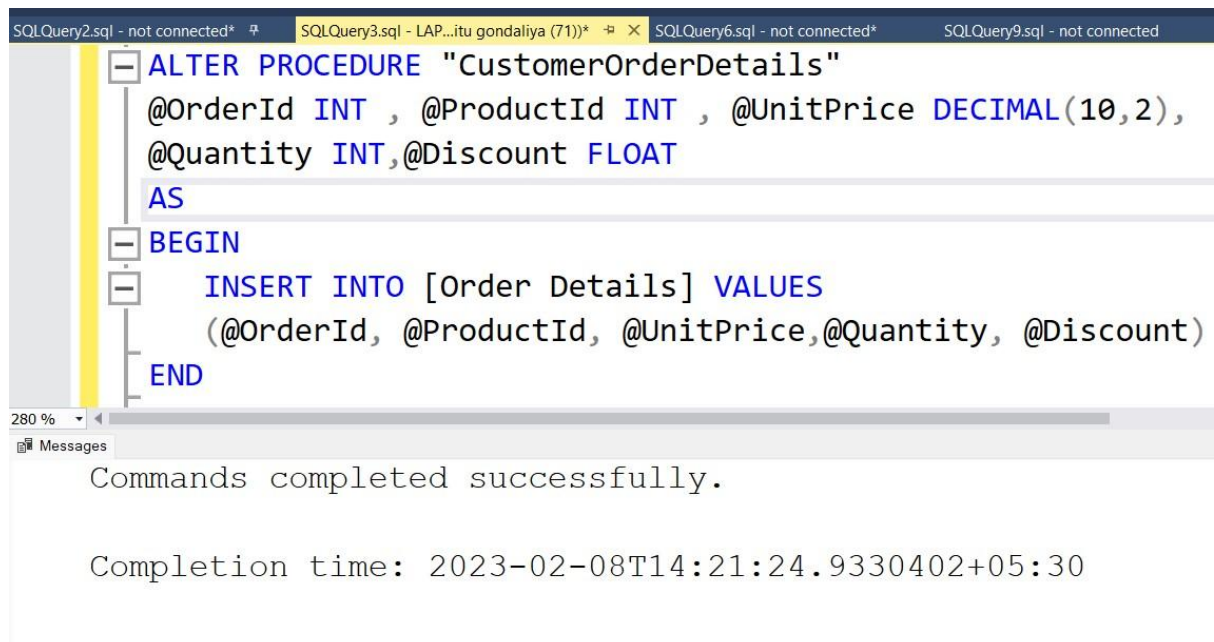
Results Messages

	TenMostExpensiveProducts	UnitPrice
1	Côte de Blaye	263.50
2	Thüringer Rostbratwurst	123.79
3	Mishi Kobe Niku	97.00
4	Sir Rodney's Marmalade	81.00
5	Carnarvon Tigers	62.50
6	Raclette Courdavault	55.00
7	Manjimup Dried Apples	53.00
8	Tarte au sucre	49.30
9	Ipoh Coffee	46.00
10	Rössle Sauerkraut	45.60

## Query 6 :

write a SQL query to Create Stored procedure in the Northwind database to insert Customer Order Details\*/

```
ALTER PROCEDURE "CustomerOrderDetails"  
@OrderId INT , @ProductId INT , @UnitPrice FLOAT,  
@Quantity INT,@Discount FLOAT  
AS  
BEGIN  
    INSERT INTO [Order Details] VALUES  
    (@OrderId, @ProductId, @UnitPrice,@Quantity, @Discount)  
END
```



The screenshot shows the SQL Server Enterprise Manager interface. The top pane displays the SQL query for creating the 'CustomerOrderDetails' stored procedure. The query is as follows:

```
ALTER PROCEDURE "CustomerOrderDetails"  
@OrderId INT , @ProductId INT , @UnitPrice DECIMAL(10,2),  
@Quantity INT,@Discount FLOAT  
AS  
BEGIN  
    INSERT INTO [Order Details] VALUES  
    (@OrderId, @ProductId, @UnitPrice,@Quantity, @Discount)  
END
```

The bottom pane shows the execution results, indicating that the commands were completed successfully. The completion time is 2023-02-08T14:21:24.9330402+05:30.

Execute CustomerOrderDetails '10248','69','10.00','3','0.6'

SELECT \* FROM [Order Details]

SQLQuery2.sql - not connected \* SQLQuery3.sql - LAP...itu gondaliya (71))\* SQLQuery6.sql - LAP...itu gondaliya (63))\* SQLQuery9.sql - not connected S

Execute CustomerOrderDetails '10248','69','10.00','3','0.6'

SELECT \* FROM [Order Details]

280 %

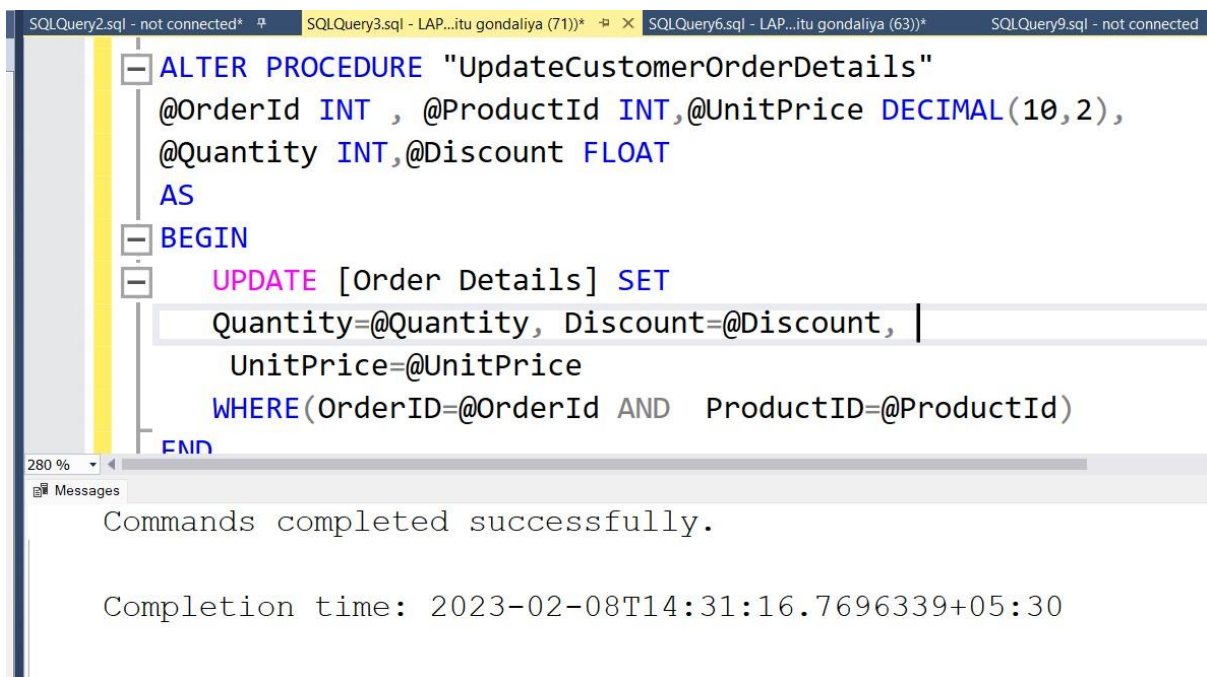
Results Messages

	OrderID	ProductID	UnitPrice	Quantity	Discount
1	10248	10	10.00	2	0.5
2	10248	11	11.00	3	0.6
3	10248	42	9.80	10	0
4	10248	69	10.00	3	0.6
5	10248	70	10.00	3	0.6
6	10248	72	34.80	5	0
7	10249	14	18.60	9	0

**\*Query 7 :**

write a SQL query to Create Stored procedure in the Northwind database to update Customer Order Details\*/

```
ALTER PROCEDURE "UpdateCustomerOrderDetails"
@OrderId INT , @ProductId INT,@UnitPrice DECIMAL(10,2),
@Quantity INT,@Discount FLOAT
AS
BEGIN
    UPDATE [Order Details] SET
    Quantity=@Quantity, Discount=@Discount,
    UnitPrice=@UnitPrice
    WHERE(OrderID=@OrderId AND ProductID=@ProductId)
END
```



The screenshot displays the SQL Server Enterprise Manager interface. The top pane shows the execution of the following SQL query:

```
ALTER PROCEDURE "UpdateCustomerOrderDetails"
@OrderId INT , @ProductId INT,@UnitPrice DECIMAL(10,2),
@Quantity INT,@Discount FLOAT
AS
BEGIN
    UPDATE [Order Details] SET
    Quantity=@Quantity, Discount=@Discount,
    UnitPrice=@UnitPrice
    WHERE(OrderID=@OrderId AND ProductID=@ProductId)
END
```

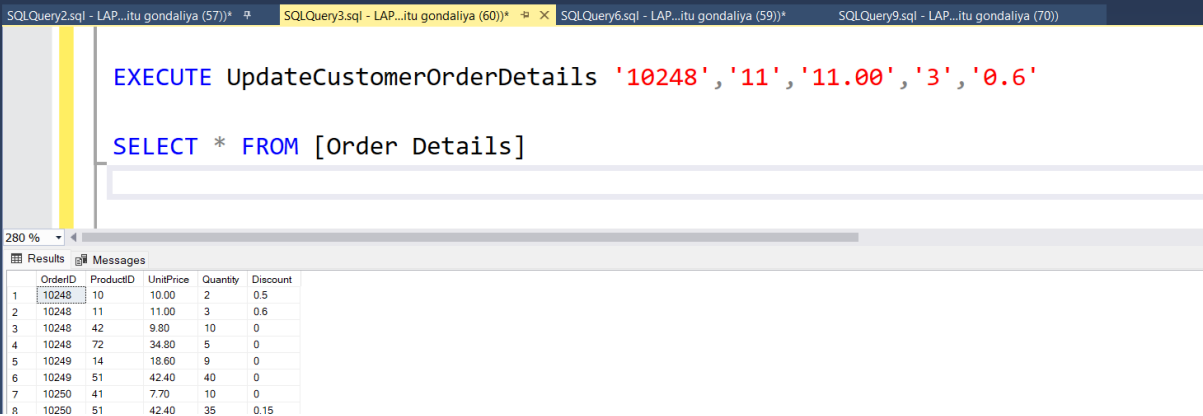
The bottom pane, titled "Messages", shows the execution results:

```
Commands completed successfully.

Completion time: 2023-02-08T14:31:16.7696339+05:30
```

EXECUTE UpdateCustomerOrderDetails '10248','11','11.00','3','0.6'

SELECT \* FROM [Order Details]



The screenshot shows a SQL Server Enterprise Manager interface. At the top, there are several tabs for SQL queries. The active tab is 'SQLQuery3.sql - LAP...itu gondaliya (60))'. Below the tabs, the query window contains two SQL statements: 'EXECUTE UpdateCustomerOrderDetails '10248','11','11.00','3','0.6'' and 'SELECT \* FROM [Order Details]'. Below the query window, there is a 'Results' pane showing a grid of data. The grid has five columns: 'OrderID', 'ProductID', 'UnitPrice', 'Quantity', and 'Discount'. The data is as follows:

	OrderID	ProductID	UnitPrice	Quantity	Discount
1	10248	10	10.00	2	0.5
2	10248	11	11.00	3	0.6
3	10248	42	9.80	10	0
4	10248	72	34.80	5	0
5	10249	14	18.60	9	0
6	10249	51	42.40	40	0
7	10250	41	7.70	10	0
8	10250	51	42.40	35	0.15